Lab 4: Pseudocode for Policy Gradient

Course staff

Algorithm 1 Actor Critic PG

INPUT: number of episodes to rollout N, learning rate for policy α, learning rate for baseline β, discount factor γ, total number of iterations E

INITIALIZE: Policy network $\pi_\phi(s)$ with parameters ϕ, baseline (value function) network $V_\theta(s)$ with parameters θ

for $e = 1, 2, 3 \ldots E$ **do**

  Rollouts

  Collect rollouts using the current policy $\pi_\phi(s)$, collect a total of N trajectories, each trajectory is a sequence $\{s_0, a_0, s_1, a_1 \ldots a_{T-1}, s_T\}$ with corresponding rewards $r_i, 0 \leq i \leq T$.

  Train Baseline

  From the collected trajectories, compute monte carlo estimate of value functions. For state s_i in a given trajectory, its estimated value function is

$$\hat{V}(s_i) = \sum_{t=i}^{T} r_t \gamma^{t-i}$$

  Compute value function loss

$$L_v = \frac{1}{N(T+1)} \sum_i (V_\theta(s_i) - \hat{V}(s_i))^2$$

  Update $\theta \leftarrow \theta - \beta \nabla_\theta L_v$

  Training policy

  (Similar as above) From the collected trajectories, compute monte carlo estimate of action value functions. For state action pair (s_i, a_i) in a given trajectory, its estimated action value function is

$$\hat{Q}(s_i, a_i) = \sum_{t=i}^{T} r_t \gamma^{t-i}$$

  Compute baseline for each state $V_\theta(s_i)$ and compute advantage $\hat{A}(s_i, a_i) = \hat{Q}(s_i, a_i) - V_\theta(s_i)$

  Compute surrogate loss

$$L_p = -\frac{1}{N(T+1)} \sum_i \hat{A}(s_i, a_i) \log \pi(a_i | s_i)$$

  Update $\phi \leftarrow \phi - \alpha \nabla_\phi L_p$

Additional hints

- To compute the gradients $\nabla_\phi L_p$ and $\nabla_\theta L_v$ correctly, one needs to construct computations using autodiff packages such as Pytorch or Tensorflow. Note that computing $\nabla_\theta L_v$ is like computing gradients for regression; computing $\nabla_\phi L_p$ is like computing gradients for classification.
- Using advantage estimate $\hat{A}(s, a)$ as a replacement to $\hat{Q}(s, a)$ reduces variance of the gradient estimator. This will make a big difference to the stability of the algorithm.
- There are more advanced techniques to PG algorithms such as trust region (e.g., TRPO, PPO) that might further elevate the algorithmic performance. However, for the purpose of this lab, if the algorithm is properly implemented, it should work well.